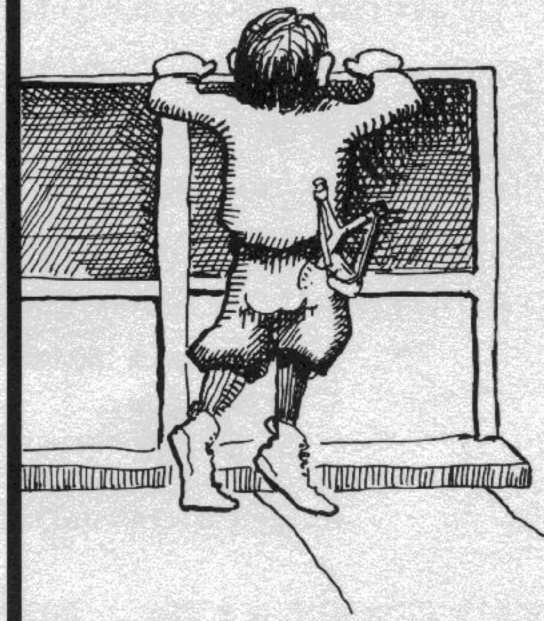
$3.00

December 1981 No. 3

TABLE OF CONTENTS

Using Modem7 4
More 4 MHz. 5
Moving Programs 5
Reverse Video Cursor 6
FORTHwords 8
Safer Formatter 12
Unica, A Set of Tools 14



REGULAR FEATURES

Letters. 2
Notes From Garland 5
Designer's Corner. 9
Book Reviews 10
Users Disk #1 15
Want Ads. 16

Source Code!

The Q/C compiler includes the full source code for a major extension to Ron Cain's Small-C:

- **For, switch/case, do-while, goto**
- **Assignment operators**
- **Improved code generation**
- **Command line arguments (argv and argc)**
- **Conditional and comma operators**
- **I/O redirection**
- **I/O library written in C**
- **Generates code for M80 (or ASM or MAC)**

Q/C does not include float, double, long, unsigned or short; static externals; initializers; sizeof; typedef; casts; structures and unions; multidimensional arrays; #ifdef, #if, #undef, #line.

For only $95 (including shipping in the US and Canada) you get the full source code and a running compiler with sample programs on disk, along with a well-written user manual. (Requires 48K CP/M system.)

We also sell CW/C, a C compiler which runs on a 56K CP/M system. It supports structures, unions, multidimensional arrays, #ifdef, and will selectively search "source library" files for functions used by your program. The I/O library for CW/C is written almost entirely in assembler. CW/C costs $75, and does **not** include source code for the compiler.

CW/C and Q/C both grew out of Small-C, but were developed independently. Jim Colvin of Quality Computer Systems implemented Q/C. We are offering Q/C for the many Small-C fans that want the source code to an extended compiler. (We still distribute the original Small-C source code on disk for only $17).

CA residents add 6% tax. Visa and MasterCard welcome.

The Code Works Box 550, Goleta, CA 93116 805-683-1585

BEST WISHES

MICRO C

FROM JAREL HAMBENNE

GIVE YOUR BIG BOARD THE TIME OF DAY

MICROTIME 80

CLOCK/CALENDAR BOARD

8 MASKABLE INTERRUPTS — 1/10 SECOND TO EVERY MONTH

FULL TIME AND DATE FUNCTIONS — TO 1/1000 SECOND

A PLUG IN REPLACEMENT FOR THE Z-80 CPU CHIP

PROGRAMMABLE INTERRUPT REGISTER

PROGRAMMABLE ALARM REGISTER

NICAD BATTERY BACKUP

COMPLETE KIT 64^{95}

ASSEMBLED & TESTED 79^{95}

Add $2.00 Postage & Handling

N.J. Residents Add 5% Sales Tax

Send Check or Money Order to:

AB COMPUTER PRODUCTS **P.O. BOX 571 JACKSON, N.J. 08527 (201) 370-9889**

MICRO CORNUCOPIA
11740 N.W. West Road
Portland, Oregon 97229
503-645-3253

Editor & Publisher
David J. Thompson

Technical Editor
Ruth Fredine-Burt

Graphic Design
Sandra Thompson

Typography
Patti Morris & Martin White
Irish Setter

Cover Illustration
Gerald Torrey

MICRO CORNUCOPIA is published six times a year by Micro Cornucopia of Oregon, 11740 N.W. West Road, Portland, Oregon 97229.

SUBSCRIPTION RATES: NEW RATES PAGE 16

1 yr. (6 issues)	$12.00
1 yr. (Canada)	$15.00
1 yr. (other foreign)	$20.00

All subscription orders payable in United States funds only, please.

ADVERTISING RATES: Available on request.

CHANGE OF ADDRESS: Please send old label and new address.

SOFTWARE, HARDWARE, AND BOOK VENDORS: Micro Cornucopia is establishing a group of reviewers. We would very much like to review your Big Board compatible products for Micro C. Please send material to Review Editor, Micro Cornucopia.

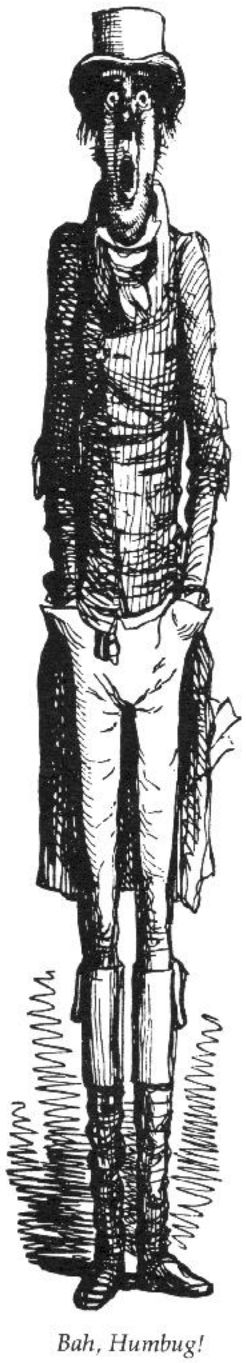
WRITER'S GUIDELINES: All items should be typed, double-spaced on white paper or better yet, on disk. (Your disk will be returned promptly.) Payment is in contributor's copies.

LETTERS TO THE EDITOR: Please sound off.

MICRO CORNUCOPIA

Dec. 1981 — The Journal of the Big Board Users — No. 3

Now We're Rolling!


Bah, Humbug!

Supporting two languages: FORTH and C.
Great news folks, we're not going to choose between Forth and C, we're going to support both. Forth should be available in ROM by the time you read this. So if you can't wait until the fourth issue for the particulars, send an SASE or call and we'll fill you in. Arne Henden is now our FORTH columnist and Hampton Miller has offered to give Arne a hand with the project. Both are professional FORTHers and both have FORTH on a PDP-11. In fact, Arne is using FORTH to control the telescopes at the Goethe Link Observatory. Now we need a volunteer or two to do a column on C.

Formatting programs.
We have now received X formatting programs where X is amazingly large and growing daily. I'm not complaining, because I know that formatting programs are not trivial. It's exciting to see the number of folks really getting into the system. I've tried all the ones I've received. One reformatted itself when I tried it, so I'm unable to pass it along. (Perhaps there's justice in there somewhere.) We'll print several of the best (see John Jones' new safe version in this issue).

Diagnostics Article.
I didn't get what I wanted from Garland this month. I had hoped that the folks at DRC would have a diagnostics article ready for this issue to help those of you bringing up stubborn boards. However, I suspect that the response to the *Byte* article still has them a little overwhelmed.

The 820.
Big Board designer, Jim Ferguson, verified that the Big Board and the Xerox 820 are, essentially identical; except that the 1771 floppy interface on the 820 can talk to either 5 inch or 8 inch disks.

Goofs
Due to mismanagement on the part of the management, we ran out of space in Issue 2 before we ran out of things we intended to run. So, we didn't get to run Andrew Beck's article on the reverse video cursor but we managed to give him credit for the direct input routine that was actually written by John Jones. We also failed to run the listing that went with Don Retzlaff's super article on moving software hither and yon. We're running that listing in this issue.

(continued on page 4)

Letters and Letters

Dear Editor,

I am a Big Board owner just trying to figure out what's going on as I practice my soldering.

What are the list prices on the books reviewed? A great volume at \$24.95 isn't as interesting to me as a good one for \$5.95.

The RAM protection circuit is great! How do I buy a "5 watt PNP"? Would you consider a \$5.00 kit?

I want to use my Big Board as soon as the solder cools, even before I invest \$1500 in discs and CP/M. Is somebody selling Tiny Basic on a 2716? How about a minimal word processor or a dumb terminal in ROM?

On the subject of disk drives, how about running a score board so folks could vote on the types of drives they like and don't like. *(Ed.—And types of disks they like and don't like.)*

When you start on the port drivers, take it easy. I don't know the Zilog mnemonics yet. Please use the comment area to explain what the instruction is doing. No, it's not obvious. Be especially careful about loops. What condition controls the branching? What bit in what register does this, and how did it get changed?

If someone out there is doing a little custom programming and EPROM burning, please let me know.

Tom Mason
Professional Engineer
2402 Audubon Rd
Akron, OH 44320

Editor's note:

"Using CP/M" sells for \$8.95. "The CP/M Handbook" sells for \$8.95. "Osborne CP/M User Guide" sells for \$12.95.

By the way, when Sig Peterson got his copy of the first issue he mentioned that the RAM protection circuit looked a little familiar. It should have, he designed it. Thanks, Sig.

The PNP power transistor for the RAM protection circuit came out of a Radio Shack PNP power transistor pack. You know, one of those 20 for \$1.99 bubble packs. None of them were marked. Like I mentioned in the article, it is probably the ultimate in non-critical applications, if it is PNP, if it works and if it looks even vaguely like the illustration then it will do fine. If the salesman really insists on a device number, you can tell him you want an MJE2901. It has a DC gain of 25, 60 V breakdown, and is rated for 10 amps and 60 watts. It is incredible overkill but it should be available for a couple of bucks just about anywhere.

As for basic in ROM, we will have something more interesting as soon as Rob gets through putting FORTH in three 2716s for the Big Board.

Dear Editor,

Thank you very much for helping me debug my system. After building the Big Board, testing memory, booting up CP/M, everything seemed to work beautifully. But then things started going wrong. Adventure would just stop, I'd tell it to "GO" and it would ask "WHERE?." One by one, my utilities gave BDOS or CRC errors which makes them unusable. How would I ever find the trouble? Then, near the end of August MICRO CORNUCOPIA arrived!!! Your article, "Power to the Big Board" said: "If the 24V supply is flaky, the drive will generate CRC errors."

The first thing I spotted was loose bolts on top of the 12,000 uF capacitor. Also, the supply had no large cap following the regulator to provide instant current. So I added 3,000 uF there. (I don't believe I'd buy another power supply made by Sunny International. It looks like it was built in somebody's garage.) Anyway, the CRC errors are gone, Hooray and thanks for the tip. Hope I can return the favor some day.

Joseph C. Kish
758 Yucca Ridge Ln
San Marcos, CA 92069

Editor's note:

Sig Peterson, a long time friend and first-rate Z80 designer, first turned me on to the strange problems caused by an inadequate +24V supply. And since he is a Big Board owner and subscriber, I'll hereby pass along your thanks to him.

Dear Editor,

It appears that I am about to conclude an agreement with a major software distributor. They will become the exclusive distributors for Timin FORTH products for CP/M after Jan 1.

So, the best I can do is offer, until Jan. 1, Timin FORTH for the Big Board for \$75.00. If will be identical with Timin FORTH release 3 except it will not include the CP/M utility package. (This package is only needed for special applications.) The visual editor will be setup for the Big Board and included on the disk.

Terms for ordering:

a. Money order or check with the order.
b. Purchaser pays the postage (usually \$2.00).
c. Californians add \$4.50 sales tax.
d. I will take no action until I receive 10 orders. If I don't receive 10 orders by Jan. 1, all the money will be refunded.

I will keep the technical hotline open as usual for these purchasers. I warrant the product to be bug-free and will supply free bug fixes if necessary.

Mitchell E. Timin
9575 Genesee Ave Suite E2
San Diego, CA 92121

Note from the FORTH editor:

We are publishing Dr. Timin's letter now to ensure that Big Board owners have time to take advantage of this excellent offer. Timin gives you fast I/O, a thoroughly debugged system, and a hotline. However, you also have several other options.

Laboratory microsystems produces Fig Forth running under CP/M for \$50 but with relatively slow I/O. Forth in ROMs will be available in a few months for about \$60. and a stand-alone Fig FORTH optimized for the Big Board with full floating point including transcendental functions will be available in the spring for around \$75.00.

The choice is yours. If you have questions call me.

Arne Henden
301-552-1295

Dear Editor,

Some comments about the Crowe Assembler. It compiles CBIOS if you delete all "-$" (it handles relative jumps without the -$ suffix, ed.) and replace all multi-byte and multi-word DEFS with single-byte and single-word definitions. Perhaps is would be a good idea to distribute a version of the CBIOS that would be compatible with this assembler.

If you get an overflow in a .RES. pseudo-op, Croweasm gives you a warning but compiles fine. You will get this warning when you compile BIAS EQU .RES. ((MSIZE-20)*1024)-200H. Note that the -200H is HEX, the "H" was left out in issue #2.

I think that exchanging programs written in C is an excellent idea.

Jan-Henrik Johansson
11124 Saffold Way
Reston, VA 22090

Editor's note:

For those of you who are interested, the original monitor and BIOS were assembled with the SD Sales assembler. One local Big Board owner purchased the SD Sales assembler from Jade Computer Products. It came on a double density disk (which he can't read) and when he asked them about it, they said that was the only way it was available. So now he is also using the Crowe assembler to do his BIOS mods. (His attitude toward the SD Sales assembler is a bit jaded right now.)
Jade Computer Products
4901 W. Rosecrans
Hawthorne, CA 90250

Dear Editor,

I was delighted when I received your first issue. In fact, I meant to write and say "Nice Job!" but somehow didn't get around to it.

Now that issue #2 is out, it's time to sharpen my pencil and tell you what I think of your publication. Here goes:

My main impression is that it's very professional. I especially like its somewhat folksy, informal flavor. Layout is attractive and easy to read.

Since your main focus is the Big Board, I'd suggest a standard box, maybe on page two, that says what a Big Board is and where to get it. Without it, the new reader might be very confused.

Your article on supporting a language is most interesting. My opinion is that FORTH should definitely not be the selection. (FORTH is not really a modern high-level language by any stretch of the imagination.) Both Pascal and C meet the criteria you mentioned in the article. One strong argument for C is UNIX. I firmly believe that UNIX will be THE operating system on the soon to arrive flood of high performance personal computers. In the near future most serious users will own a personal computer with a million bytes of RAM and a 16 or 32 bit processor (like the 68000).

With that sort of processing power, a system with the limitations of CP/M simply won't survive. Sure, 64K Z80 systems will be around for a long long time but when a full-blown 68000 system with a megabyte of RAM is available for $2,000, we'll all start using UNIX or one of its imitators.

As you know, C is the language that UNIX (and all its related applications software) was written in. This all seems like a compelling argument for adopting C as your "most favored" language. (But since we are the developers of the CW/C compiler, I guess this whole line of reasoning is predictable.)

Again, good luck with your venture. I don't own a Big Board, but I still very much enjoy your magazine.

Ron Jeffries
The Code Works
Box 550
Goleta, CA 93116

Editor's note:

Thanks for the kind comments, Ron. Regarding information about the Big Board, we've had an amazing number of questions about the system from folks who see the magazine but who aren't familiar with the system. So Jim Tanner will be running his famous "Everything he could cram into a single page ad" ad, beginning in issue #4.

As for UNIX, I know what you mean. I am using a UNIX-like set of tools on my system now (called UNICA) and now that I have them, I wonder how I got along without them. On the other hand, C stands very well on its own merits. For applications ranging from simple machine control to business and scientific programs, C is hard to beat. Plus, it has had a standard pretty much from the beginning which has gone a long way toward making the source code transportable.

I've also noticed that a surprising number of the heavy FORTHers I've met, are either into C as their second language or are very interested in it. (The reverse hasn't necessarily been true.) Although they are implemented quite differently, both languages provide the programmer easy access to the hardware.

Dear Editor,

The conversion to double density requires several modifications, both to the hardware and to the software.

1. The system must be running reliably at 4 MHz.
2. The Western Digital 1771 chip must be replaced by either the Western Digital 1791 or better the 1795 chip set.
3. The monitor has to be changed (we are using two EPROMs).
4. The bios has to be able to deblock the larger sectors to CP/M's 128 byte sectors.
5. The floppies have to be formatted for D-Density.
6. A new type Sysgen is required.

4 MHz modification.

If you are having trouble making other 4 MHz modifications work, try the following:

Change the crystal from 20 Mhz to 16 MHz. Remove and discard U 97. Change U 77 from 74LS04 to 7404. Cut trace from U 96 pin 4, and cut trace from U 76 pin 4 and 5 then connect U 76 pin 3 to trace from pin 4, and connect U 76 pin 4 to trace from pin 5. Run jumper from U 96 pin 5 to U 97 pin 8, run jumper from junction of R 38 & R 40 (pin 9 of U 77) to U 96 pin 5, and run jumper from U 96 pin 4 to U 97 pin 9.

(continued next page)

And More Letters

During a phone conversation, Mr.Tanner noted that several of his customers need to use 5 inch floppies. So we are designing a new board which will handle double sided drives, and 5 inch floppies. (It will be ready by December 15th.) However, it will be a while after that before we have software support for the 5 inch and double sided drives.

Otto Hiller
P.O.Box 1294
Madison, WI 53701
608-271-4747 3-5pm

Dear Editor,

I am using a Heathkit H-19 with my Big Board and have designed a smaller monitor since the H-19 is doing most of the display work.

I have also modified the disk formatting program so that you can format disks even though you only have one drive.

Gary Mion
408-241-1766 home
408-987-5079 work

Editor's note:

Gary is doing some very interesting things with the H-19 and the Big Board. Considering the number of folks who have contacted me about using the H-19 with the system, this should be welcome information.

Gary's formatting program is on user disk #1 (FORMAT3.COM).

Anyone interested in what Gary is doing is welcome to call him at the above numbers.

■ ■ ■

Using Modem7

By Andrew Beck

AB Computer Products
PO Box 571
Jackson, NJ 08527

Modem7 is a general purpose public domain modem program available from the CP/M users group and on our user disk #1. The following patches for the MODEM7 program will allow it to run on the Big Board.

```
Port and bit mask setups:

ADDR    DATA
0103    00      Not PMMI modem.
0104    00      No front panel.
0105    00      Use FF if you're running 4 MHz.
0106    FF      Use 00 to delete same name files.
0107    00      Use FF to come up in expert mode.
010A    07      Port B status (A=06)
010D    05      Port B data (A=04)
0110    04      Xmit ready bit.
0113    04      Xmit ready mask
0116    05      Port B data (A=06)
0119    01      Rec ready bit
011C    01      Rec ready mask

Initialize the SIO port B for 8 bits/character.

ADDR    DATA
14E4    3E      Reset SIO errors
14E5    F0
14E6    D3
14E7    07
14E8    3E      Select SIO reg. 3
14E9    03
14EA    D3
14EB    07
14EC    3E      8 bits/char
14ED    C1
14EE    D3
14EF    07
14F0    3E      SIO register 5
14F1    05
14F2    D3
14F3    07
14F4    3E      Xmit 8 bits/char
14F5    EA
14F6    D3
14F7    07
14F8    3E      Set baud rate gen B
14F9    XX      See pg 8 in Theory of Op
14FA    D3      to select value of XX
14FB    0C
14FC    C9      Return from initialization
```

■ ■ ■

(Editorial continued)

Raising subscription rates.

Sandy figured out what it is costing us to print and mail the magazine and the annual total per subscriber is so close to $12.00 it's scary. We're doing everything we can to save money, everything, that is, but scrimp on the printing.

Take a look at the difference in print quality between issues #1 and #2 and you'll see why we're so delighted to have a real craftsman printing Micro C. His name, by the way, is Dave Cramer.

Two weeks after we figured out our costs, the post office changed them by raising postal rates. So, in response to reality, we are announcing new rates for the new year. See page 16 for details. (Sigh!)

Disks.

If you send us material on disk, and you don't mind if we keep the disk in your file (our original 100 disks are almost used up) then print "FILE" followed by your name and address (phone number) on the label. Otherwise print "RETURN TO . . ." and name, etc.

Christmas.

Here it is October and I'm having to write about Christmas. Well, I hope your Christmas and New Year are as positive and as exciting as you've made this past year for us.

[illegible]

David Thompson
Editor and Publisher

More 4 MHz

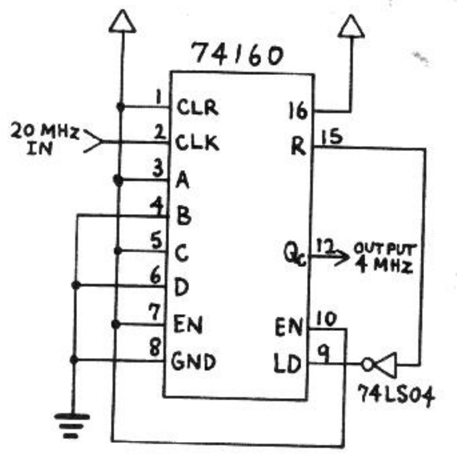
This issue continues the saga of the speeding Big Board. You MUST have a fast monitor ROM. The standard 2716 which is a 450 ns part just won't make it. You need a 2716-1 (350 ns). If you don't have access to a ROM burner or the fast parts, drop us a note here at Micro C and we'll see if we're set up to help you yet.

Another Mod
By Philip Atkinson

The following modification replaces U96 (72LS293) with a 74160 and a spare gate on a 74LS04. This results in a 45% duty cycle.

The 13 ns propogation delay of the inverter steals from state 5 and adds to state 9. After the system is running 4 MHz, locations F115H and F119H should be changed to 125 (base 10) to maintain a 1 second interrupt.

Schematic of 4 MHz generator
The 74160 replaces U96



■ ■ ■

Notes from Garland, Texas

By David Thompson

I talked to Ditigal Research about the problems people were having getting the board to run 4 MHz and they suggested:

If you have trouble making your system run after making the latest 4 MHz modification, verify that the following parts are "A" (4MHz) parts, Z80-A, PIO-A, SIO-A, and CTC-A. Plus, you probably need to change the PFM monitor ROM to a 2716-1. The 1771 and the 300 ns RAM should work but you could check them as a last resort.

They are working on a modification to a minifloppy interface (AB Computers) so that one system can interface with both 5 and 8 inch drives. This way you could easily transfer files between 5 and 8 inch disks. He also mentioned that the 5 inch interface depends on the 1771's clock separator rather than the more dependable external separator used for the larger disks. He is trying to make the external circuitry work for 5 inch also. ■ ■ ■

Moving Programs

By Don Retzlaff

```
*****************************************
*                                       *
*  EXAMPLE OF PROGRAM TO INSTALL PROGRAM IN
*          UPPER MEMORY                 *
*                                       *
*          DON RETZLAFF                 *
*                                       *
*****************************************
*
        PSECT ABS
        ORG   0FA00H        ;FINAL PROGRAM LOCATION
*
**      EQUATED SYMBOLS
*
PATCH   EQU   0FxxxH        ;PATCH ADDRESS
*
**      ROUTINE TO PUT PROGRAM IN UPPER MEMORY
*
MOVEUP  EQU   $             ;SETUP POINTERS
        LD    HL,INIT-$+100H
        LD    DE,INIT
        LD    BC,LENGTH
        LDIR                ;MOVE PROGRAM UP
*
**      INITALIZE ROUTINE
*
INIT    EQU   $
        LD    BC,START      ;GET ADRS OF ROUTINE START
        LD    (PATCH),BC    ;PUT ADRS IN PATCH LOCATION
        RET                 ;RETURN TO CPM
*
**      YOUR PROGRAM
*
START   EQU   $
*
**
**
***** YOUR PROGRAM INSERTED HERE!!!
**
**
*
        RET   ;RETURN TO CPM OR CALLING ROUTINE
*
LENGTH  EQU   $-INIT
        END
```

end

Reverse Video Cursor

By Andrew Beck

AB Computers
PO Box 571
Jackson, NJ 08527

One of the most common means of highlighting characters on a video terminal is by "inverting" the character, that is producing a black character on a white background. Character highlighting can greatly improve the appearance of data on the screen by making certain characters stand out. It just so happens that the most popular cursor employed is also of the reverse character type.

As supplied by Digital Research, the Big Board does not incorporate either reverse character highlighting or a reverse character cursor. Here, I will show you how a few minor changes to the Big Board can produce both of these.

The cursor as implemented on the big board is either an underline character for blank spaces, or a blinking character. Lets look at the schematic and see how this blinking character is produced and what changes have to be made to convert it to a reverse video character.

How it works. U21 generates a 4 Hertz square wave at pin 8. Pin 12 of U37 is connected to bit 7 of the video display RAM, and is ANDed with the 4 Hz square wave by U37 pins 12 and 13.

As the memory is scanned by U47 and U50, pin 11 of U37 will be high for each character that has it's bit 7 set, during the time that pin 8 of U21 is also high. This is the "blink" clock. This clock is fed to U25 pin 2 where it is ORed with the output of U12. The result is then ORed with SC3.

This causes pin 6 of U25 to go high if either SC3, pin 9 of U12 or pin 11 of U37 goes high. Since pin 6 of U25 is connected to the chip select pin of the character generator PROM, this causes the outputs of the PROM to tristate and produce a blank character field.

Therefore, setting bit 7 of a character will cause it to blink at a 4 hertz rate. You may be wondering what the purpose of the SC3 signal and the output of U12 are. Well, these two signals cause the character generator to blank out between character lines and while the CPU is accessing video memory.

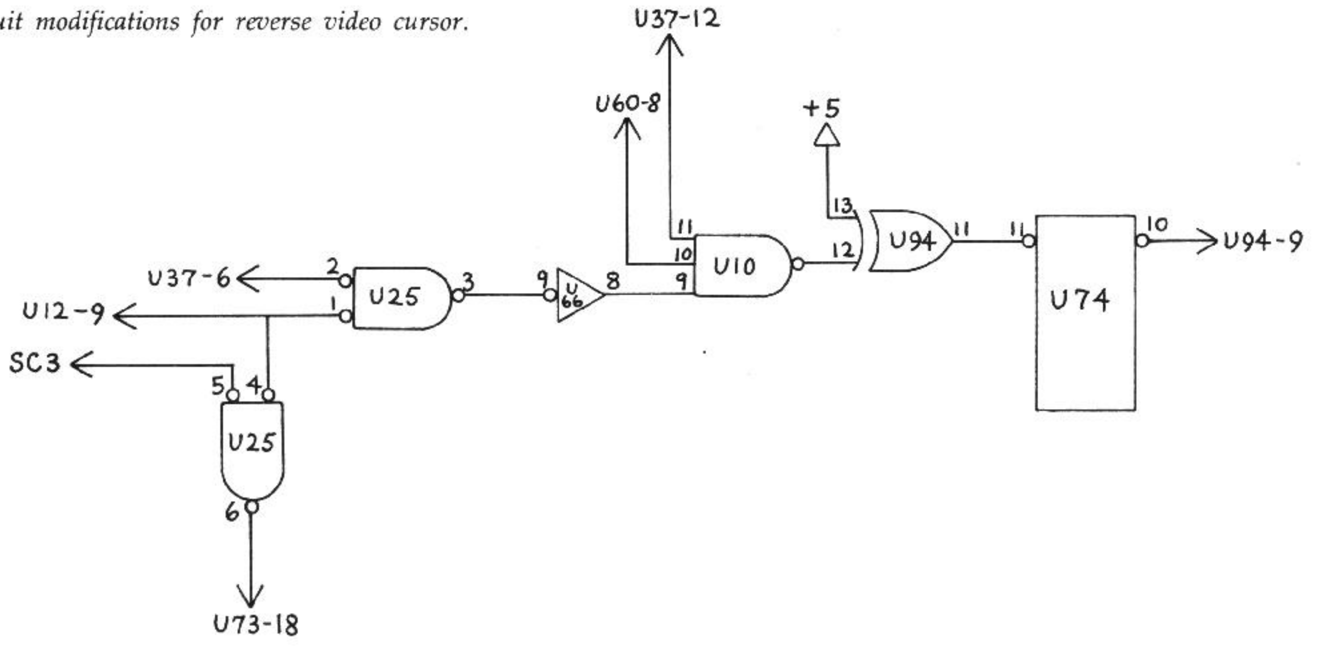
Changes to the Big Board. Now that we know how to produce a blinking character we can look at the mods necessary to produce an inverted character.

Inverted video is generated by "inverting" the video signal, that is making it a NOT of itself. If we make this happen selectively on characters with bit 7 set we can generate a reverse video cursor as well as reverse video highlighting. Let's trace the partial schematic shown and see how the modification works.

First we cut the connection between U25 pin 3 and 4 and connect pin 4 to U25 pin 1. This prevents bit 7 of the character RAM from affecting the character PROM while still allowing the CRTE and SC3 signals to cause the PROM to blank the character. U10 pins 9, 10 and 11 AND the vertical retrace and the horizontal retrace with the output of U66 pin 8. This is the NOR of the CPU access and bit 7 of the character RAM. U94 then inverts this signal and feeds it to pin 11 of U74. If bit 7 of a character is set, a high will be clocked into U74 when the character is addressed if the CPU is not accessing display RAM and the vertical and horizontal retrace signals are not low.

Now that we have a signal which will go high only when the addressed character's bit 7 is set, it should be a simple matter to make it invert the character. Part of U94 is used to determine the logic level of

Circuit modifications for reverse video cursor.



the video signal fed to your monitor. By connecting pin 10 of U74 to pin 9 of U94 we will produce an inverted character whenever bit 7 is set.

If you would normally connect JB1 10 and 11 together then omit the connections to U94 pins 11 and 12 and connect U10 pin 8 directly to pin 11 of U74.

If you want your cursor to be a reverse blinking box then connect pin 11 of U10 to pin 11 of U37 instead of pin 12.

We now have the hardware to generate reverse video but PFM will still use an underline character for a cursor at blank spaces. Below is a listing of the area of PFM that must be changed to produce a reverse video cursor all the time.

I am writing a new CRT driver for PFM and will submit it for publication in this magazine when it's done. Not only will it provide the reverse video cursor but it will also allow your programs to produce reverse video characters.

```
Board Modifications

Cut the following runs:
(All are on the bottom
of the board.)

U25 pin 4 to U25 pin 3
U25 pin 2 to U37 pin 11

Jumper the following:

U25 pin 4  to U25 pin 1
U25 pin 2  to U37 pin 6
U25 pin 3  to U66 pin 9
U66 pin 8  to U10 pin 11
U10 pin 9  to U37 pin 11*
U60 pin 8  to U10 pin 10
U10 pin 8  to U94 pin 12
U94 pin 13 to U94 pin 14
U94 pin 11 to U74 pin 11
U74 pin 10 to U94 pin 9

*Use pin 12 instead for
a non-blinking cursor.
```

Changes to CRT driver in monitor.

This listing represents the area of the CRT driver module that must be changed.

```
;*********************************************************
;*                                                       *
;*      MEMORY-MAPPED CRT OUTPUT DRIVER                  *
;*                                                       *
;*      Russell Smith     18-August-1980                 *
;*                                                       *
;*********************************************************
;
;
CRTBAS  EQU     CRTMEM.SHR.8        ;STARTING PAGE# OF 3K CRT SPACE
CRTTOP  EQU     CRTMEM+3072.SHR.8 ;ENDING PAGE# OF CRT SPACE
;
;
CRTOUT: PUSH    HL
        PUSH    DE
        PUSH    BC
        RES     7,A
        LD      C,A
        DI                          ;KEEP THE WOLVES AWAY FOR A WHILE
        LD      (SPSAVE),SP
        LD      SP,TMPSTK+32        ;POINT SP TO TOP OF LOCAL STACK
        IN      A,(BITDAT)
        SET     7,A                 ;SELECT ROM/CRT MEMORY BANK
        OUT     (BITDAT),A
;
;       FIRST REMOVE THE OLD CURSOR CHARACTER FROM THE SCREEN
;
        LD      HL,(CURSOR)         ;LOAD HL WITH CURSOR POINTER
        LD      A,H
        AND     00001111B           ;A LITTLE INSURANCE THAT HL CAN'T
        OR      CRTBAS              ; EVER POINT OUTSIDE THE CRT MEMORY
        LD      H,A
        RES     7,(HL)              ;REMOVE CURSOR BY DISABLING INVERT
;
;       PROCESS CHARACTER PASSED IN C
;
        CALL    OUTCH
;
;       NOW STORE A NEW CURSOR CHARACTER AT THE CURSOR LOCATION
;
        SET     7,(HL)              ;THEN TURN ON BIT 7 TO ENABLE INVERT
        LD      (CURSOR),HL         ;SAVE HL AS CURSOR POINTER

        LD      SP,(SPSAVE)
        IN      A,(BITDAT)
        RES     7,A                 ;SWITCH BACK THE LOWER 16K OF RAM
        OUT     (BITDAT),A
        EI                          ;INTERRUPTS ARE SAFE AGAIN
        POP     BC
        POP     DE
        POP     HL                  AREA CHANGED
        RET
;
;
OUTCH:  LD      DE,LEADIN
        LD      A,(DE)              ;GET LEAD-IN SEQUENCE STATE
        OR      A
        JP      NZ,MULTI            ;JUMP IF IN A LEAD-IN SEQUENCE
        LD      A,C                 ; ELSE PROCESS CHARACTER IN C
```

■ ■ ■

FORTHwords

Column by Arne A. Henden

7415 Leahy Road
New Carrollton, MD 20784

My first taste of FORTH came as an astronomy graduate student at Indiana University. We had purchased an IMSAI 8080 microcomputer in 1976 to automate the position readout of the 36-inch reflecting telescope of Goethe Link Observatory. We spent 8 months of effort (the first 5 months using paper tape I/O before being able to buy a floppy disk drive) creating a 5000-line, 8K byte assembly language program that calculated sidereal (star) time and the direction that the telescope was pointed.

There just had to be a better way! This early in the game, though, we found only interpreted BASIC supported on CP/M, and it was just too slow for our needs. However, at a meeting in 1977, we heard that the University of Rochester had adapted FORTH to the 8080, calling their version URTH. As FORTH was well-known in the astronomical community as a great instrument control language (and because we got it gratis), we implemented URTH on our system as a stand-alone operating system.

Soon we were controlling the telescope in FORTH, taking one month to code in FORTH what took us eight months in assembly language. On a 16K byte system, we had enough room left over to start thinking about letting the IMSAI control data acquisition, and to perhaps automate the entire telescope movement. We achieved these goals this past year in 32K bytes of memory.

In 1979 I moved to Washington, D. C. to work for the Goddard Space Flight Center, and am currently maintaining on a PDP-11/44 minicomputer a version of Cal-Tech FORTH that I heavily modified. We use FORTH for image processing of astronomical imagery, to analyze other astronomical data. In the near future we'll use it as the operating system on an LSI-11/23 to control a PDS 1010A microdensitometer (used to digitize photographic plates).

From my experience, FORTH is the only language to use for instrument control, for systems with minimal memory, and for experimenters making new hardware interfaces. It is the fastest interactive language available. As most of you have built your own computer, you are more interested in the insides of a computer than someone who purchased an Apple or TRS-80, and are more likely to enjoy using FORTH. It will be my goal to convince you of this!

What is FORTH?

FORTH is a programming language that is ideally suited to the microcomputer environment. It is an interpreter, allowing you to write routines, debug them, and throw them away without ever having to leave FORTH. It is a compiler, because each new definition that you enter is changed into a list of the addresses of each FORTH word that you called upon in your routine. When the definition is subsequently executed, FORTH calls each address like a subroutine. It is an assembler, so that you can write routines directly in machine language and link them to other routines written in higher-level FORTH. In other words, FORTH combines aspects of an operating system, an assembler, and a high level language into one.

FORTH starts with about 40 primitives, or basic definitions, that are written in machine language. All later operations are defined in terms of these basic words. A standard version of FORTH will have around 100 secondary definitions in addition to the 40 primitives mentioned above. These secondary words consist of a list of addresses of previously defined words. When a secondary is executed, FORTH will call each of the routines whose addresses are in the list. This linked list structure creates an extremely compact language. Written in FORTH, an assembler, editor, disk and terminal I/O will all fit in 16K bytes with plenty of leftover room for user applications. This makes FORTH ideal as a standalone operating system supporting a single user.

There are no argument lists in FORTH. Instead, parameters are passed on a data stack, common to all words. Of course, named variables, constants and arrays are also available so that conventional high-level programming can be used. However, all of these named storage regions are FORTH words in themselves, and you can easily create new data types such as matrices or multiple precision floating point variables.

FORTH is a structured language, supporting conditional branching and loop structures, but GOTO statements are not allowed. Each defined word is usually short, no more than 3 or 4 lines of source code. This creates a highly modular program structure that is exceptionally easy to debug. I find the incremental compilation especially useful in writing machine language routines. I can compile, debug and correct routines several times before I can do one assembly using ASM or MAC.

There is no faster interactive language. For non-floating point operations, high-level FORTH runs 10-20 times faster than interpreted BASIC. Yet, because of the easy inclusion of machine language routines, time-critical applications can be divided into fast acquisition definitions overseen by slower but higher-level commands.

However, all does not smell like roses in the FORTH garden. It is a new language and has only recently undergone standardization. Programmers now agree on about 150 basic definitions, but these are usually only the core of most available versions of FORTH. The added definitions are of course non-standard.

Contrary to some statements made in print, FORTH is not as fast as truly compiled languages such as the Microsoft BASIC and FORTRAN compilers. There is considerable overhead in secondary definitions. A FORTH function would take 2-3 times as long to run as a similar function in optimized, compiled FORTRAN.

Because the stack is used heavily for argument list passage, it is easy to make errors. In addition, since stack parameters have no labels, clarity of programming suffers. Comments are often left out of definitions because of their compactness. FORTH is known as a language that is easy to use but hard to read.

Some data types, such as floating point, are not normally supported. The FORTH-79 standard does not include floating point; however, you are certainly not prevented from including your own in software or with an arithmetic processor such as the AM9511.

There is no typing of variables. By this I mean that once the number is on the stack, the system doesn't know whether it is to be a byte, integer, double precision integer, or whatever. Each operation, such as multiply, is defined for a specific type of variable, such as 16-bit integers. The programmer must keep track of what values are on the stack and use the appropriate operator.

FORTH is a really excellent language for instrument control. You can access all memory locations directly, input and output bytes from ports, set interrupt vectors and all other internal functions with ease. It is easily made into a multi-user system, requiring about 2K bytes additional programming. We used a standalone FORTH at GSFC on a PDP-11/40 with 64K bytes of memory that handled three users simultaneously: 2 performing mundane operations such as tape copying and spectral analysis, and one using the image processing system.

FORTH is not suited for number crunching. A compiled language, such as FORTRAN, PASCAL, or C would be a better choice.

Versions of FORTH.

FORTH is a new language. It was designed around 1970, and has recently undergone standardization at meetings in 1977, 1978 and 1979. The 1978 meeting produced a version of FORTH implemented by the FORTH Interest Group called Fig-FORTH. It was to be a test case for the new standard, and met with instant acceptance (the fact that it was distributed by a non-profit organization at little cost helped!). The 1979 meeting made minor changes to the Fig-FORTH model, publishing their resultant vocabulary in a document entitled "FORTH-79." Therefore, you will generally find three versions of FORTH on the market.

Fig-FORTH. Distributed by the FORTH Interest Group, P. O. Box 1105, San Carlos, CA 94070. There is no Z80 version, only an 8080 one. FIG only offers the source listings at $15 each, along with an installation manual for another $15. You must enter the program by keyboard. Be warned that the 8080 source listing has over 4000 lines of code.

FORTH-79. Most F-79 systems that you will see advertised are Fig-FORTH that has been modified to match the standard. FIG does not offer F-79 FORTH at this time.

pico- and polyFORTH. FORTH, Inc. (where Charles Moore, the originator of FORTH, lives) has their own versions of FORTH on the market. PicoFORTH ($500) is an entry level system designed to whet one's appetite for polyFORTH ($2500). While embracing the FORTH-79 standard, polyFORTH is greatly expanded and offers multiuser and target compiling features. Check with FORTH, Inc. at 2309 Pacific Coast Highway, Hermosa Beach, CA 90254 for more details on these two products.

How do I get started?

I suggest that you buy the book, *Starting FORTH*, and get used to the language. A good supplier of all FORTH literature and some software packages is Mountain View Press (MVP), P. O. Box 4656, Mountain View, CA 94040. They advertise in BYTE every month.

Also, the BYTE issue of August, 1980 was devoted to FORTH. This back issue can be purchased from MVP or from FIG. If you are still interested in FORTH, then you have three options.

(1) Buy *Threaded Interpretive Languages* and design your own FORTH. I only recommend this to die-hard programming types.

(2) Buy a copy of Fig-FORTH for the 8080. The source listing can be obtained from FIG and entered if you are willing to sit at a keyboard for two weeks. An alternative is to buy Fig-FORTH already on disk from MVP for $65. It sure beats typing if you value your time at all.

(3) Buy a commercial FORTH. There are several on the market, with prices ranging from $50 to $2500. I will be reviewing several of these in the next issue, and suggest that you hold off any purchases until then. We may have a little surprise at that time.

Future columns

If you read the book reviews in this issue and buy the book that meets your needs, then you will be just about ready to read the language reviews next time.

In addition, future projects on back burners include: a plotting package for the Epson and Paper Tiger printers, dark room applications, FORTH in ROMs, Startrek, Sargon, and EPROM programming.

This column is for you. So if you have any thoughts about or answers for projects, find some aspect of FORTH confusing, have questions to be answered, or just want to sound off, drop me a line. I will be happy to help.

■ ■ ■

Designer's Corner

Keyboard bit 7. There is an ungrounded rumor going around that keyboard bit 7 must be tied low. Not so! The first character sent from the keyboard must have bit 7 low so the PFM monitor can tell there is a keyboard out there, but that's it.

The rest of the time, bit 7 is treated just like any normal bit, getting passed on to the data bus just like all the rest. This should be interesting to anyone with special function keys. You can distinguish these special keys by setting bit 7 and then, if necessary, have the Z80 reprocess them.

Reviews of FORTH Books

By Arne Henden

Threaded Interpretive Languages
by R. G. Loeliger
Byte Books 1981
243 pages + index
$18.95 hardcover

When I first saw the advertisement for *Threaded Interpretive Languages* (TIL).in BYTE, I thought that it would be a book about standard FORTH. While the particular TIL that Loeliger discusses (called ZIP) is very similar to FORTH, it is not FIG-FORTH or FORTH-79 standard. Loeliger tries to be very general and to discuss TILs as a group of programming languages. He indicates where the infrequent differences between ZIP and FORTH occur, and continually offers alternatives to standard definitions so that the reader can customize his implementation to meet his personal needs.

This text is not for the casual FORTH user. For example, no examples of keyword (FORTH routine) use are presented, and the 2-block editor is the only FORTH block-oriented listing given. The book discusses the Z80 only, and requires a Z80 mnemonic assembler. The text is for someone who is considering writing his own FORTH or, having purchased and used a commercial FORTH, is interested in how it ticks.

Chapters 1 through 3 (38 pages) describe what a TIL is, and gives basic flowcharts of how to write one.

Chapter 4 (35 pages) lists those keywords that Loeliger feels are necessary in the implementation of a TIL. For example, arithmetic operators, DO-LOOPs, and conditional branching constructs such as IF-ELSE-THEN are discussed.

Chapters 5 and 6 (106 pages) are the heart of the TIL implementation. The construction of the outer interpreter is shown, along with all primitive routines such as the number converter and cold/warm restart routine. Chapter 6 is very useful in that Loeliger lists each keyword in alphabetical order, its function, input/output (I/O) requirements, how used, and then shows sample Z80 assembly source for the keyword.

Chapter 7 (54 pages) describes three extensions to the basic TIL: a Z80 assembler, disk I/O, and an editor. The assembler is particularly well presented and is one of the best that I have seen. The disk I/O and editor are described in general terms, with important keywords mentioned but not described in full detail. A line-by-line editor is given in a listing. A short chapter (10 pages) on TIL usage completes the text, followed by a bibliography and brief index.

Aside from the 3 or 4 typographic errors that I found, there are no obvious errors in *Threaded Interpretive Languages*. However, Loeliger could have added an index for all of the keywords discussed, and could have included a master listing for his implementation on ZIP. There are no worked problems at the end of each chapter, because this is a manual on how to construct a TIL, not how to use one. The text was prepared in a very professional manner, with good illustrations and liberal use of italics and enhanced type to emphasize pertinent points. I highly recommend this book for the advanced assembly language programmer. Even if you own a commercial FORTH, Loeliger presents new definitions that may not be present in your version (such as an assembler). This text plus a user's manual would make a very complete reference library for FORTH.

■ ■ ■

Invitation to FORTH
Harry Katzan, Jr.
Petrocelli Books, Inc. 1981
222 pages + index
$18.50 hardcover

To the best of my knowledge, this was the first hardcover book that purported to cover FORTH. Katzan has tried to produce a text similar to most high level language tutorials (such as for FORTRAN): that is, as processor independent as possible. His primary audience are persons relatively unfamiliar with computers or other languages.

The text is photo-offset from dot matrix printer output. While the quality of reproduction is good, the printer does not have lower case descenders. Because of this printer limitation, special characters and underlines have to be hand drawn and as a result, there are many errors.

Most of these errors are very obvious and should have been caught by the author. It gives the impression that Katzan rushed this book to press.

Invitation to FORTH is a limited introduction to the language. It covers only a subset of Fig-FORTH and FORTH-79 words, leaving out system related words such as BUILDS or WORD. There is no coverage of any assembler or editor vocabularies.

Chapter 0 (11 pages) contains a brief introduction to FORTH. Following this there are three chapters of background material (63 pages) that for the most part could have been left out.

Chapters 4-9 (132 pages) cover all of the high level FORTH definitions. Katzan has broken the words into straightforward categories and presents numerous examples of how each word is used. Some of the subjects such as the difference between constants and variables are nicely presented.

Chapter 9 ends abruptly, and there is no final chapter that draws everything together. The text is followed by references, answers to exercises, and an index.

The only good point of *Invitation to FORTH* is the use of many examples interspersed with the text. By contrast, the exercises at the end of each chapter are very dull and were not written with much thought.

Again, the lack of descenders makes the text difficult to read for any length of time, and the numerous errors really hurt, especially when they occur in examples. I recommend that you pass over this book and buy something else.

■ ■ ■

Starting FORTH
Leo Brodie
Prentice-Hall 1981
348 pages + appendices
\$16 softcover, \$20 hardcover

I have read just about everything on FORTH that I can get my hands on. This includes so-called beginner's manuals, system manuals, and even the three manuals that I've written. Believe me, *Starting FORTH* stands head and shoulders above them all. Brodie has created a real masterpiece: a book in a simple conversational style that will keep beginners interested, and yet with enough meat that the experts will find something to learn.

While the book covers FORTH-79 standard definitions, Brodie is employed by FORTH, Inc. and you will find many added definitions that are only available in their product line. Brodie was very careful to indicate which words were not FORTH-79, and in many cases why they disagree with the standard.

Brodie's style is conversational, very much as if you had a personal tutor. Since FORTH is a vocabulary-oriented language, Brodie has tried to make FORTH come to life. There are many illustrations, most of which are in a form not often seen in textbooks: cartoon characters.

For example, a Frenchman interpreter, a masked executioner, and a numbers-runner are used for the text interpreter, the EXECUTE word, and NUMBER respectively. While some readers may object to the third-grade level of this kind of illustration, they get the point across without being so dry.

There are footnotes liberally included for both beginners and experts. Each word defined is listed in the section where it is introduced, and also in a summary at the end of every chapter. There is a review of terms and a set of exercises at the end of each chapter. The problems are all well thought out and are fun to solve. For example, problem 7 at the end of chapter 2 states:

"You're the inventory programmer at Maria's Egg Ranch. Define a word called EGG.CARTONS which expects the number on top of the stack to be the total number of eggs laid by the chickens today and prints out the number of cartons that can be filled with a dozen each, as well as the number of left-over eggs."

Chapters 1 and 2 (55 pages) cover elementary FORTH: what the stacks are, how to define simple words, and basic arithmetic and stack operators. Brodie spends four pages explaining how the text interpreter works and an equal space on the power of a stack. Almost all examples are presented in FORTH form, in word explanations, and also in diagrams.

Chapter 3 covers a basic FORTH editor. Commands are included to operate on lines and strings. Having used several FORTH systems, I can say that the names of the commands may differ from one FORTH system to another, but the functions performed are almost always the same. There is no standard for editors at this time. Brodie also presents words to list, load, copy and save FORTH block buffers and explains how to remove definitions from the dictionary.

Chapters 4 and 6 (36 pages) cover conditional branching structures such as IF-ELSE-THEN and also loop structures such as DO-LOOP. The branching conditions are shown as segments of railroad track, with switching to control the program path.

Chapters 5 and 7 (52 pages) cover fixed point arithmetic and number formatting. Brodie makes the point that most operations can be performed in integer arithmetic with appropriate scaling.

Brodie primarily covers FORTH, Inc. methods of defining words (count field with 3 unique characters). In code and parameter areas, he uses FORTH, Inc methods. Your system may be different.

Still, the basic concepts are the same, and you will find little difficulty in tracing your FORTH. Explaining the innards of FORTH can be very difficult, and Brodie has done about as well as anyone that I've seen.

Chapter 10 (34 pages) covers basic input/output commands from both disk and terminal. Chapter 11 (27 pages) is another system level chapter, covering how you create new defining words and compiling words to extend the flexibility of FORTH. Brodie shows how to create a word that will define arrays, and presents a simple flowcharting method for FORTH.

The last chapter (32 pages) covers three examples of FORTH programming: a word game, a basic file system, and mathematically calculating the weight of a pile of material. These examples are well documented, and are excellent to tie the text together. Four appendices cover answers to all exercises, polyFORTH enhancements, FORTH-79 differences, and a summary of all FORTH words defined in the text.

I only have two negative comments about *Starting FORTH*. There is no overall index, making it difficult to locate a given subject or where the text discusses the use of a basic definition. Also, polyFORTH is mentioned too often, primarily because this is a text designed for FORTH, Inc. products.

In all, I highly recommend *Starting FORTH*. It is by far the best manual available.

■ ■ ■

Safer Formatter

By John Jones
5826 Southwest Ave.
St. Louis, MO 63139

When PFM-80 attempts to select a drive, it first checks for drive ready. If the requested drive is NOT ready, the previously selected drive remains active. With the original FORMAT program this means that the disk in drive A will be formatted!

This is a new version of the diskette formatter which not only corrects this bug, but also loops back to a prompt message. The program will now allow additional disks to be formatted and gives you the chance to change your mind and abort the format before it starts. (This formatter is called FORMAT2.COM on user disk #1.)

```
        TITLE   'SSSD DISKETTE FORMATTER '
;
;       THIS PROGRAM IS DESIGNED TO FORMAT A SINGLE DENSITY
;       SINGLE SIDED SOFT SECTORED 8" DISKETTE INTO STAN-
;       DARD 128 BYTE SECTORS.  DESIGNED TO RUN ON THE
;       FERGUSON BIG BOARD Z-80 COMPUTER, IT TAKES AD-
;       VANTAGE OF THE WD-1771 FLOPPY DISK CONTROLLER'S
;       CAPABILITIES FOR SEMI-AUTOMATIC FORMATTING.
;
;
;       WRITTEN: J.P. JONES  4/20/81    MODIFIED: J.P.J. 5/14/81
;       PROMPT MESSAGES ADDED: J.P.J. 8/15/81
;
;
        ORG     100H            ;STD CP/M COM PROGRAM
;
BOOT    EQU     0               ;CP/M BOOT
MONITR  EQU     0F000H          ;WILL USE SOME OF PFM-80
WDSTAT  EQU     10H             ;1771 STATUS ADDR
WDCTL   EQU     WDSTAT          ;CONTROL = STATUS WRITE
WDATA   EQU     13H             ;1771 DATA I/O
HOME    EQU     MONITR+1EH      ;DISK HOME ROUTINE
SEEK    EQU     MONITR+21H      ;SEEK TRACK ROUTINE
SELECT  EQU     MONITR+1BH      ;SELECT DRIVE ROUTINE
ENTRY   EQU     5               ;CP/M ENTRY POINT
GETCHR  EQU     1               ;GET CONSOLE CHARACTER FUNCTION
PRTSTG  EQU     9               ;PRINT STRING ON CONSOLE
CTLC    EQU     3               ;^C
CR      EQU     13              ;
LF      EQU     10
CRLF    EQU     LF*256+CR       ;WHY USE TWO DEFB'S?
;
;
;       PROMPT WITH INSTRUCTIONS AND ALLOW EXIT
;
START   LD      DE,PROMPT       ;POINT TO MESSAGE
        LD      C,PRTSTG        ;CP/M FUNCTION #, PRINT STRING
        CALL    ENTRY
;
        LD      C,GETCHR        ;GET KEYBOARD ENTRY
        CALL    ENTRY
;
        CP      CTLC            ;ABORT?
        JP Z    BOOT            ;IF ^C, YES
;
;
;       FIRST, SET UP ONE TRACK'S DATA IMAGE
```

```
;       DO THE TRACK WRITE
;
        LD HL, LEADER           ;POINT TO DATA
        LD D, 20                ;20 * 256 + 36 = TOTAL BYTES
        LD B, 36
        LD C, WDATA             ;C POINTS TO 1771 DATA PORT
        LD A, 0F4H              ;WRITE TRACK COMMAND
        OUT (WDCTL), A          ;SEND COMMAND
;
NXTBYT  HALT                    ;WHEN 1771 READY, WILL NMI
        OUTI                    ;SEND BYTE
        JP NZ   NXTBYT
        DEC D                   ;OUTER BYTE COUNTER
        JP NZ   NXTBYT
;
;       DO SETUP FOR NEXT TRACK
;
        POP     BC              ;GET TRACK COUNTER BACK
        INC C                   ;UPDATE IT
        LD A,C
        CP 77                   ;IF 77, DONE
        JP NZ   NXTTRK
        POP AF                  ;GET BYTE BACK FOR NMIVEC
        LD (66H), A
        EI                      ;REENABLE NORMAL OPERATION
        JP      START           ;FIND IF NEED TO FORMAT ANOTHER
;
;
PROMPT  DEFW    CRLF
        DEFM    'Place diskette to be formatted '
        DEFM    'into drive B'
        DEFW    CRLF
        DEFM    'Press space bar to start,'
        DEFM    ' ^C to quit.'
        DEFW    CRLF
        DEFB    '$'
;
NOTSEL  LD DE,  NOBMSG
        LD C,   PRTSTG
        CALL    ENTRY
        JP      START
;
NOBMSG  DEFW    CRLF
        DEFM    'Error in selecting drive B'
        DEFW    CRLF
        DEFB    '$'
;
;
;       TRACK DATA FOLLOWS
;
LEADER  DEFW    -1              ;40 BYTES OF FF
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        LD HL, DATA             ;POINT TO DATA AREA OF DISK IMAGE
        LD DE, DATA+1
        LD (HL), OE5H           ;E5 = BLANK
        LD BC, 127
        LDIR                    ;FILL DATA AREA
;
;       COPY ONE SECTOR 25 TIMES
;
        LD HL, SECT1            ;START OF SECTOR 1 DATA
        LD DE, SECT2            ;ADD ONTO END
        LD BC, 186*25           ;186 BYTES PER SECTOR
        LDIR                    ;FILL THAT MEMORY
;
;       NOW SET UP SECTOR NUMBERS IN PROPER SLOTS IN TRACK IMAGE
;
        LD HL, SECTNO           ;POINT TO SLOT IN FIRST IMAGE
        LD DE, 186              ;OFFSET TO NEXT SECTOR
        LD BC, 26*256+1         ;B=CTR, C=SECTOR #
;
SECTID  LD (HL),C               ;STORE SECTOR #
        INC C                   ;INCREMENT SECTOR #
        ADD HL,DE               ;POINT TO NEXT SECTOR DATA
        DJNZ    SECTID          ;DO ALL 26
;
;       PUT TRAILER OF FF'S AFTER WHOLE TRACK
;
        LD HL, SECT1+4836       ;POINT AFTER DATA
        LD A, -1
        LD B,0                  ;DO 256 BYTES FOR INSURANCE
ENDMRK  LD (HL), A
        INC HL
        DJNZ    ENDMRK
;
;       SET UP TO FORMAT DISK IN DRIVE 'B'
;
        LD C, 1                 ;B IS DRIVE #1
        CALL    SELECT
        OR A                    ;TEST FOR SELECT SUCESS
        JP NZ   NOTSEL          ;UNSUCESSFUL, DON'T FORMAT!!
;
;       DO SETUP FOR FORMATTING
;
        LD A,(66H)              ;GET BYTE AT NMI VECTOR
        PUSH    AF
        LD A, 0C9H              ;RET INSTRUCTION
        LD (66H), A             ;STORE RETURN
;
;       DO THE FORMAT
;
        DI                      ;CANNOT INTERRUPT
        LD C,0                  ;START WITH TRACK 0
NXTTRK  LD B, 26                ;26 SECTORS PER TRACK
        PUSH    BC              ;NEED THEM LATER
;
;       SEEK NEXT TRACK IN SEQUENCE
;
        CALL    SEEK
;
        POP     BC              ;TRACK # AND SECTOR CTR BACK
        PUSH    BC              ;WILL NEED AGAIN
;
;       PUT TRACK ID'S IN PROPER PLACE IN TRACK IMAGE
;
        LD HL, TRKNO            ;POINT TO POINTER IN IMAGE
        LD DE, 186              ;OFFSET FOR EACH SECTOR
TRAKID  LD (HL), C              ;STORE CURRENT TRACK NO.
        ADD HL,DE               ;POINT TO NEXT SECTOR
        DJNZ    TRAKID
```

```
        DEFW    -1
        DEFW    -1
        DEFW    -1
;
        DEFW    0               ;6 BYTES OF 0
        DEFW    0
        DEFW    0
;
        DEFB    0FCH            ;WRITE INDEX MARK COMMAND
;
        DEFW    -1              ;26 BYTES OF FF
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
SECT1   DEFW    0               ;TRACK DATA STARTS
        DEFW    0               ;6 BYTES OF 0
        DEFW    0
        DEFB    0FEH            ;WRITE ID ADDRESS MARK COMMAND
TRKNO   DEFW    0               ;FIRST BYTE OF WORD = TRACK #
SECTNO  DEFW    0               ;FIRST BYTE OF WORD = SECTOR #
        DEFB    0F7H            ;CRC COMMAND
;
        DEFB    -1              ;11 BYTES OF FF
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    0               ;6 BYTES OF 0
        DEFW    0
        DEFW    0
        DEFB    0FBH            ;WRITE DATA ADDRESS MARK
DATA    DEFS    128             ;ACTUAL DATA AREA
;
        DEFB    0F7H            ;WRITE CRC COMMAND
        DEFB    -1              ;27 BYTES OF FF
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1
        DEFW    -1              ;END OF ONE SECTOR
;
;       ON EXECUTION THE PRECEDING SECTOR DATA WILL BE
;       DUPLICATED 25 TIMES TO GIVE A FULL 26 SECTORS
;       OF DATA.  THE 256 BYTES FOLLOWING THE TRACK
;       DATA WILL BE FILLED WITH FF's WHICH FILL THE
;       AREA BETWEEN THE LAST SECTOR AND THE NEXT INDEX.
;       THE WD-1771 EXITS THE 'WRITE TRACK' MODE ON
;       RECEIPT OF THE INDEX SIGNAL.
;
SECT2   END
```

end

Unica, A Set of Tools

Review by David Thompson

When you think of software you usually think of compilers, assemblers, applications programs; perhaps a disassembler. You don't usually think of simple task doers that make life with a computer a lot easier. That's what Unica is, a workbench full of fast, easy to use, Unix-like utilities. Now that I've had them on my system disk for a while, I'm really dependent on them.

The files.

bc.com—binary file compare. Reads two files as binary data and displays their differences in 16 byte blocks.

cat.com—concatenate files. This routine reads each declared file in sequence and then outputs to the screen or other file etc.

EXAMPLE cat *.txt *.doc >lst:
Print (lst:) all the .txt files followed by the .doc files.

EXAMPLE cat -v a:*.txt b:*.doc >>articles.doc
Concatenate all the .txt files on drive A followed by all the .doc files on drive B. Verify (-v) with the operator each file before it's included. Append (>>) the result to articles.doc on the default drive.

cp.com—copy files. This is a fancy copy routine that, for instance, copies a memory full of data at a time when moving large blocks of data. It can also do things like change ownership of a file or refuse to copy a file if one of the same name already exists on the destination drive.

dm.com—disk mapper. Depending on selected options, this routine generates a map of the disk and/or a detailed summary of the disk use.

fid.com—file identification by a 16-bit CRC. fid.com computes a 16-bit cyclic redundancy character for each file in its argument list. If the CRCs of two files are the same, chances are about 65K to 1 that the files are the same.

hc.com—horizontal concatenator. That's right! I didn't make this one up, Andy really wrote a horizontal concatenator. It pastes the lines from one file onto the ends of the corresponding lines from another file (or another two files . . .). It even does some useful things like breaking up lines inside a file into chunks short enough for even the most finicky editor. It also does some other very useful things but I'll let you discover them for yourself.

ln.com—link a new name to a file. This is a good routine for multiuser environments or where you are writing software for novice and experienced users. For example, you could have a SUBMIT.COM file that also had the name S.COM. The .SUB file could have a descriptive name and a single letter name. Thus, the novice could be instructed to simply type S C <CR>.

ls.com—list directory. This is a large program (14K) for simply listing a directory but you can get a lot of information (there are 12 flags for this routine). For instance this routine will tell you how many bytes in each file on a disk (to the byte) and it does the count and output in about the same time it takes DIR to just display the filenames.

mv.com—move (rename) a file. At first, you'd wonder why anyone would bother with another rename utility, after all we have CP/M's RENAME. On the other hand, I use this one a lot.

EXAMPLE mv -n v1*.* v2*.new
Make all the filenames which begin with v1, begin with v2 and make all their extensions into .NEW.

sc.com—source file (text) comparator. This routine provides a really first class display. Where the two files differ it displays the lines that differ in two blocks, one from each file. Each of the two blocks ends in the same line. If the two files are the same, it generates no output.

sfa.com—set file attributes. This lets you set all .COM files, for instance, to read only (or remove read only status).

sp.com—spelling error check. Compared with most of the spelling routines on the market, this routine alone is worth the price of admission.

I got 17K worth of text off the local community bulletin board service (CBBS) and I ran the spell program from Unica and the spell program from The Word. Unica came up with 155 suspect words, The Word found 55. (For perspective, The Word is the best spelling program on the market, period. I'll be reviewing The Word in issue #4.)

There were actually 5 misspelled words, the rest were names or strange abbreviations. On a piece of straight text, a larger percentage of the suspect words found by both programs would be real misspellings. Unica's most significant problem here is it doesn't yet let the user create a custom dictionary. (At least not yet.)

Example SC.COM output Comparing two versions of Small C+

```
C>sc a:cc1.c b:cc1.c
----- a:cc1.c line 23 ---------------
#define BANNER  "Small C Plus       version AO 1.10,  Mar 1981"

/* System-dependent parameters */
----- b:cc1.c line 23 -----
#define BANNER  "Small C Plus       version AO 1.11,  May 1981"

/* System-dependent parameters */
----- a:cc1.c line 687 ---------------
        int sq[6];                  /* local queue */
        sq[lqsym]=locptr;           /* record local level */
----- b:cc1.c line 687 -----
        int sq[6],*ptr;
        sq[lqsym]=locptr;           /* record local level */
----- a:cc1.c line 701 ---------------
        zpop();                     /* clean up stack */
----- b:cc1.c line 701 -----
        ptr=readloop();
        label(ptr[lqinc1]);
        zpop();                     /* clean up stack */
```

sr.com—search files for a pattern. The speed of XM-80 really shines through here (not to mention the algorithm). This routine took 1 minute and 30 seconds to search for the string "cie" in 30 files containing 80K of text. It found with word "sufficient" and displayed the filename, and the line.

srt.com—sort lines in alphabetic order. I've used srt as a filter after wx so that I get an alphabetic list of all the words I've used, along with a count of the times I've used each one. (It also works great with lines of source code.)

tee.com—pipe the input to two outputs.

wc.com—word counter. This routine calculates a character, word, and line totals for all the files in its argument list. For the input "wc *.*," the routine would display the totals for each file on the disk.

wx.com—extract words. Turns a file into a series of one-word lines. The output of this routine can be piped through srt.com and then to the printer or another file etc.

Note: I have barely touched on the capabilities of each of these tools. when used alone. Unica's thorough and very well organized manual really covers the individual commands. However, even it is only able to suggest a few of the multitude of possibilities for simple command-line combinations to do very powerful things.

Conclusion.

Unica, by itself, is the best bargain in software I've seen. The quality of the user interface, the documentation, and code is absolutely first class. When combined with XM-80, this is a set of assembly language programming tools that is untouched anywhere.

In a future issue we'll look at XM-80 (extended mac-80), the language in which Unica is written. By itself, Unica is available for $95. If you include XM-80 it's $195 from:

Andrew Klossner
Knowlogy
PO Box 283
Wilsonville, OR 97070

■ ■ ■

Users Disk Number 1

Because of incredible interest, we are making user disk #1 available. The price is $15.00, which includes disk and postage.

However, if you send two disks, one containing one or more articles and one blank, we'll put user disk #1 on the blank disk and return it to you free. The disk containing your contribution(s) will be filed in a folder with your name on it, and your material will be catalogued for use in the magazine and/or on user disk #2.

Any article or piece of software you've written for the system qualifies you for the free copy. However, it's best to check with us about the subject so we don't get three or four of the same thing.

Contents of User Disk Number 1.

COPYALL.COM quickly copies a disk track for track (including tracks 0 and 1) onto the object disk you specify. If there is anything on the object disk, it just gets written over. The routine is very fast and also verifies what it has written by reading the object disk and comparing what it reads against memory. After reporting an error to the screen, the program continues the transfer. These copy programs are from the CPMug.

COPYFAST.COM does the same thing except it does not include tracks 0 and 1.

CPLUS.MAN is the manual for Small C+. I was going to put Small C+ on this user disk since it is public domain. But the folks at Alpha Omega who did the extensions to Small C said they would sell Small C+ for $25 (instead of $50) if they could support the language through Micro Cornucopia. So it's now available from:

Alpha Omega Computer Systems Inc.
PO Box U
Corvallis Oregon 97330

The price is $25.00 post-paid for an 8" CP/M disk containing the compiler, the source for the compiler (in Small C), the documentation, and some demonstration software. The compiler generates 8080 assembly code which must be assembled with M80 and linked with L80.

Since this is public domain software, we can write utilities, add features, etc. and pass them around the group along with small C+ itself. This way it can become a group project for those interested in hands-on compiler experience (the compiler is written in Small C and compiles itself). Also, software written in Small C+ is compatible with the full blown versions of C.

CROWEASM.COM. This program is designed to assemble the Z80 instruction set using standard mnemonics defined in the Zilog Z80 Assembly Language Programming Manual. If you don't have this book you probably should pick it up.

Rules for writing the source code are as defined in the manual except that only upper case characters are recognized for labels, operators and operands. All defined pseudo-operations are implemented except MACRO, ENDM, COND, and ENDC; i.e. no macro instructions or conditional assembly. All assembly time functions are implemented, plus two borrowed from Intel: .HIGH. and .LOW. The object code output is in absolute format only.

FORMAT2.COM. This is an update by John Jones of his original FORMAT.Z80. It will not format drive A if it can't format a disk in drive B. It prompts you to insert a disk in B and if for some reason it can't format that disk it simply repeats the prompt. ^ C gets you back to CP/M.

FORMAT3.COM & FORMAT3.Z80. This is another update of the FORMAT.Z80 program. Like FORMAT2.COM, it has been made safe to use. It also does a track by track verification, displaying a V for each good track and an N for each bad track.

Minor Bug—If you accidentally enter a letter other than A, B, C, or D, for the drive select you might select a drive you didn't expect. (pretty remote chance though). This mod was done by Gary Mion.

OTHELLO.COM was originally from the CPMug but since it is one of the few games that I occasionally have time or inclination to play, I thought I would include it.

PR.COM and P.COM are little pieces of software I wrote to do two things. First they modify the BIOS slightly to make sure that a character sent to the display in the C register returns in the A register. The PFM monitor doesn't bother to do this and some programs (including adventure) require it.

Second, they put the lst: device on serial port B at 9600 baud and they stuff two nulls (00) every time they see an ASCII LF or CR. These characters are inserted simply to flush the SIO.

MODEM7.COM, MODEM7.ASM, MODEM7.DOC. This is a modem program from the CPMug. See Andrew Beck's modifications for MODEM7 in this issue.

WANT ADS

The following folks are reaching you for only 20 cents per word. If you would like to reach the same audience, send your words and 20 cents for each, to Micro Cornucopia, 11740 NW West Rd., Portland, OR 97229.

Big Board owners. Enhance your system with a professional, lightweight, painted structural enclosure. Adequate in size to enclose the Big Board, power supply, fan and keyboard of your choice. Plus strong enough to support a 17" monitor on top. Custom AC power-I/O rear panel simplifies installation and wiring for your application or ours. Further information, write Microvisions P.O. Box 2371, Woburn, MA 01801

Wanted - I need the following issues of Interface Age magazine: Dec 75, Jan 76, Feb 76, Mar 76, Sept 76, Dec 76, Mar 77, & Apr 77. Will pay top dollar, any condition. Call Andy collect at 201-370-9889 days or 201-370-9568 evenings.

Looking for other Big Board owners in your area? Run a WANT AD in Micro C.

New Subscription Rates

Starting January 1, 1982, the annual subscription rate will be:

1 yr. (6 issues) U.S.	$16.00
1 yr. Canada, Mexico	$20.00
1 yr. Other Foreign	$26.00

Back issues, each:

U.S., Canada, Mexico	$3.00
Other foreign	$5.00

Folks in the U.S. can get their issues sent first class by paying the Canadian rate.

Hear Yea! Hear Yea!

A Modest Proposal

As is the ancient custom and spirit of the season, we at Micro Cornucopia are asking our patrons to send money. And, taking a page from other great beggers of our time (Wards, Sears, Hickory Farms . . .) we are offering something of modest value in return.

Super Deluxe Package

Can't think of anything for those very special people in your life who've been bugging you for something really different? Then get them complete Big Board kits and subscriptions to Micro Cornucopia. It'll keep them busy for months. (It kept you busy for that long didn't it?)

Sorta Deluxe Package

If the Super Deluxe Package seems a little spendy how about getting a bare board and Micro Cornucopia? Let the ingrates scrounge up their own parts.

Basic, No Frills Package

So you called Digital Research Computers and they told you it would be six weeks before they could ship even a bare board, your number-one person seems to be trying for number-two, your cat died, and the court won't let you claim bankruptcy two Christmas' in a row. What do you do? Order a gift subscription to Micro Cornucopia and let that special person buy a Big Board for him or herself.

What else could you give for $12.00 that would either get them on the stick or remind them of what they are missing for a full year? We'll rush issue #3 out first class (or for an additional $6.00 we'll include issues #1 and #2) with a nicely hand-lettered note that says Merry Christmas from you. That way they'll know that you're really not an ill-mannered, thoughtless slob after all. (Plus, you beat the January 1 price hike.)

So fill out a subscription form for each person on your list. Fill it out just as they would, but at the very top, write "Gift From" followed by your name.

Thanks and Happy Holidays,

Dave

Fa La La La La La La La La

ANNOUNCING

DOUBLE DENSITY DISK INTERFACE FOR THE BIG BOARD

New floppy interface package for the Big Board lets you read and write single and double density disks with 128, 256, 512, and 1024 Bytes/sector.

The package includes:

1. Fully assembled and tested board, cable and connector to replace the 1771. Board contains 5 ICs including a Western Digital 1795.

2. An extended monitor in two 2716s.

3. A disk containing:
   - Disk formatting program. (128, 256,512,1024 bytes/sector)
   - Disk copying program.
   - An overlay for MOVCPM.COM
   - A double density SYSGEN

4. Documentation
   - Dependable 4MHz mod
   - Jumpers to add to the Big Board.

Sector size is determined by how the disk was formatted and is totally transparent to the user.

Disk capacity ranges from: 241K for SS, SD, 128 bytes/sector to 668K for SS, DD, 1024 bytes/sector

Requires minor modification to Big Board and requires that Big Board run 4 MHz.

Available December 15, 1981.

Price: $200 before February 15th.
       $240 after February 15th.

Otto Hiller Co.
Scientific Equipment
Box 1294
Madison, WI 53701

Bring the flavor of Unix To your Z80-based CP/M system with Unica

"Unicum: a thing unique in its kind, especially an example of writing. Unica: the plural of unicum."

The Unica: a unique collection of programs supporting many features of the Unix operating system never before available under CP/M. The Unica are more than software tools; they are finely crafted instruments of surgical quality. Some of the Unica are:

bc	-	binary file compare
cat	-	catenate files
cp	-	copy one or more files
dm	-	disk map and statistics
hc	-	horizontal file catenation
ln	-	create file links (aliases)
ls	-	directory lister
mv	-	move (rename) files, even across users
rm	-	remove files
sc	-	source file compare, with resynchronization
srt	-	in-memory file sorter
sr	-	search multiple files for a pattern
sp	-	spelling error detector, with 20,000 word dictionary

Each Unicum understands several flags ("options" or"switches") which control program alternatives. No special"shell" is needed; Unica commands are typed to the standard CP/M command interpreter. The Unica package supports several Unix-like facilities, like filename user numbers:

```
        sc data.bas;2 data.bas;3
```

(compares files belonging to user 2 and user 3);
Wildcard patterns:

```
        rm *tmp* -v
```

(types each filename containing the letters TMP and asks whether to delete the file);
I/O redirection:

```
        ls -a ►list
```

(writes a directory listing of all files to file"list");
Pipes:

```
        cat chap* ! sp ! srt ►lst:
```

(concatenates each file whose name starts with"chap", makes a list of mispelled words, sorts the list, and prints it on the listing device).

The Unica are written in XM-80, a low level language which combines rigorously checked procedure definition and invocation with the versatility of Z80 assembly language. XM-80 includes a language translator which turns XM-80 programs into source code for MACRO-80, the industry standard assembler from Microsoft. It also includes a MACRO-80 object library with over forty "software components", subroutine packages which are called to perform services such as piping, wildcard matching, output formatting, and device-independent I/O with buffers of any size from 1 to 64k bytes.

The source code for each Unicum main program (but not for the software component library) is provided. With the Unica and XM-80, you can customize each utility to your installation, and write your own applications quickly and efficiently. Programs which you write using XM-80 components are not subject to any licensing fee.

Extensive documentation includes tutorials, reference manuals, individual spec sheets for each component, and thorough descriptions of each Unicum.

Update policy: each Unica owner is informed when new Unica or components become available. At any time, and as often as you like, you can return the distribution disk with a $10 handling fee and get the current versions of the Unica and XM-80, with documentation for all new or changed software.

The Unica and XM-80 (which requires MACRO-80) are priced at $195, or $25 for the documentation. The Unica alone are supplied as *.COM executable files and are priced at $95 for the set, or $15 for the documentation. Software is distributed on 8" floppy disks for Z80 CP/M version 2 systems.

Knowlogy
"Shaping Knowledge for Evolving Worlds"
P.O. Box 283
Wilsonville, Oregon 97070

Visa/Mastercard customers call (503) 635-5701 after hours for next day shipment.

CP/M is a trademark of Digital Research; Unica is a trademark of Knowlogy; Unix is a trademark of Bell Telephone Labs; XM-80 is a trademark of Scientific Enterprises; Z80 is a trademark of Zilog Inc.

MICRO CORNUCOPIA
Journal of the Big Board Users Group

P.O. BOX 223
BEND, OREGON 97709